# THE USE OF MULTICAST AND INTEREST MANAGEMENT IN DIS AND HLA APPLICATIONS

**Edward T. Powell**
**Science Applications International Corporation**

**KEYWORDS**
Multicast; Interest Management, Filtering, HLA, STOW

## ABSTRACT

It was recognized early on in the DIS standardization process that the use of interest management and multicast could add significantly to the scalability and thus usefulness of the DIS protocol. Interest management is the process of each application expressing interest in the type of data it needs and then receiving (through some infrastructure mechanism) only that information requested. It is clear that multicast can be the "infrastructure mechanism" that is used to carry interest-managed data. Despite the widespread discussion about interest management and multicast, only a handful of multicast schemes have ever been discussed in the DIS literature. By far the most popular scheme has multicast groups being assigned on the basis of geographic grids. The next most popular scheme has multicast groups being assigned on the basis of PDU type. This paper begins by reviewing all of the multicast usage schemes that have previously been presented in the DIS literature. The paper then presents a more systematic view of the use of multicast in a DIS context, categorizes the possible types of multicast schemes into three basic categories (source-based, data-based, and destination-based), explores the interest management requirements of each type of scheme, gives a number of novel examples of the use of multicast, examines the role of network hierarchy in each multicast usage example, and goes on to describe a multicast usage scheme that was proposed for use in the STOW program. Finally, because no one scheme is useful for all types of exercises, the paper suggests a number of actions that can be taken by the DIS/HLA community to experimentally examine the different multicast schemes and develop an approach for being able to use multiple filtering and multicast algorithms in future HLA RTI applications.

## 1.0 INTRODUCTION

When DIS was first invented, it was generally realized that it would be unsuitable for large-scale exercises because of the way DIS Protocol Data Units (PDUs) were delivered (via broadcast). Essentially, as the total number of entities in an exercise increases, the number of PDUs per second incident on any individual simulation also increases until the simulation is so overwhelmed by incoming PDUs that it simply can't operate as intended. Luckily, most of these PDUs are irrelevant to the operation of any particular simulation. So to fix the problem, a technique was required such that only those PDUs that really mattered to a simulation were delivered to it. If such a technique could be discovered and implemented it was hoped DIS could become a scalable protocol. Thus was born the concept of *interest management*, the delivery of data based on simulations' expressed interests.

The most promising method for making interest management work is through the use of *multicast* (MC). Multicast is an addressing scheme (most prominently associated with the TCP/IP family of protocols), in which a single address stands in for a number of hosts. Thus by subscribing to a multicast group (or channel) a host receives all information that any other host transmits to that particular address.

When multicast began to become a technical reality, the DIS community expended a lot of effort trying to explore ways in which multicast could be used to enhance DIS's scalability. Numerous ideas were put forward, and numerous problems with multicast implementations were found. For instance, most networking systems could only support a limited number of MC groups at any one time (from a handful to perhaps a hundred at first; up to 1,000 or so now). Also, the time it took to join and/or leave a MC group was significantly longer (seconds to tens of seconds) than interaction times on the synthetic battlefield, leading to a number of problems implementing some of the suggested multicast schemes.

This paper is designed to be both a review of previous multicast ideas and a presentation of some new ideas and suggestions. Section 2.0 presents a historical summary of all the multicast ideas presented to the DIS community so far. Section 3.0 presents a categorization of multicast schemes along with some interesting new ideas for the use of multicast. Section 4.0 discusses the use of hierarchy and its relationship to the new schemes. Section 5.0 expands on a particularly promising idea for using multicast and Section 6.0 suggests directions for further research. Section 6.0 also describes a general way to incorporate interest management and multicast into the High Level Architecture (HLA) Run-Time Infrastructure (RTI).

## 2.0 HISTORY OF DIS MULTICAST IDEAS

The first description of a scheme for using multicast in a DIS exercise that I can find is a paper by Johnson and Myers [1] who suggest using different multicast groups depending on PDU type, Exercise ID, Radio Frequency, and geographic grid. For the geographic grid, the terrain is segmented into rectangular geographic regions called Areas of Interest (AOIs). Each entity transmits on the multicast group corresponding to the AOI in which it resides, and listens to those MC groups corresponding to areas in which it is interested. They also suggest a hierarchical approach of using "filter servers" to mediate routing of data between simulation nodes. Each server would infer the interests of the entities under its purview.

Sherman and Butler [2] also suggest segmenting the battlefield into geographic areas of interest. They go on to suggest that topographically-relevant AOIs might be more effective than regular grids.

Clay [3] suggests assigning an entity's PDUs to a multicast group sometimes on the basis of the unit it belongs to and sometimes on the basis of geographic location, although no algorithm is given to decide under what circumstances which method should be used. The author does not describe how subscription decisions would be made, leaving this scheme not fully specified.

Doris [4], summarizing the thinking of the community up to this point, describes and analyzes a number of uses of MC. He states that the primary use of MC would be to segment traffic by Exercise ID to allow multiple exercises to play without interference on the same network. He goes on to describe a number of variations on the geographic grid idea, including the concept of a two-level hierarchical grid system that would allow hosts to manage fewer MC groups if they wish, but at the cost of multiple transmissions of data. Doris then introduces the idea of filtering on the basis of entity type (which is called class-based filtering in HLA terminology). He concludes by analyzing the effect of filtering based on PDU type. After studying the effect of each of these schemes, Doris concludes that MC assignment based on AOI geographic grid system "appears to be the most viable."

In the CASS minutes from the 9th DIS Workshop [5], a number of techniques for MC usage are presented that have subsequently played a large role in the STOW Program and in the prototype HLA RTI. Geographic grid MC is discussed, and then a very significant new idea, arising out of the ARPA $10^4$ Program, is presented by Van Hook: each entity transmits on its own MC group and low-rate summaries are broadcast to give receivers the knowledge they need for subscription. Analysis of this new scheme showed that it can be a more effective filtering mechanism than the geographic grid scheme.

Van Hook *et al.* [6] present an analysis of the geographic grid scheme in which the effect of grid size is measured. Smaller grids give better filtering than larger grids, but require more total MC groups, which is not surprising. This paper also first introduced the idea of dynamically assigning MC groups, so as to conserve MC addresses for when they are needed.

A year later, Van Hook *et al*. [7], writing for the STOW team, describe the techniques used for STOW-E to achieve large scale. Because of the limitations of the DSI at the time, multicast was not used extensively, but the important idea of *hierarchical filtering* was introduced to the community in full form in this paper. Though mentioned in passing in previous papers (such as [1]), the idea of having a gateway computer filter traffic based on interests and thus essentially mediate

between the simulations on a LAN and the simulations in the rest of the world was presented. Though not directly related to the use of multicast, hierarchy is an important concept that will be discussed at length below.

In the same proceedings, Calvin and Van Hook [8] expand on the idea of using grid-based relevance filtering using an application gateway, and introduce the concept of an *agent*, software that acts on behalf of simulations in performing some task.

Van Hook *et al.* [9] go on to describe relevance filtering in detail, and explain two approaches: grid-based and object based. Grid-based multicast is by now well understood, but the authors include ideas on other types of grids, such as, "hexagonal cells, non-uniform grids, adaptive grids, three-dimensional grids, and different grid systems for each different sensor modality or filtering parameter." Object-based multicast is described as a technique whereby *relevancy tests* are defined and evaluated by subscription agents acting on behalf of simulations. The subscription agents then inform the simulations which MC groups to subscribe to. The tests can be arbitrary but the subscription agent requires some global knowledge (via a low-rate broadcast of the entire state of the simulated world) to evaluate them. The authors simulate the effects of both grid-based and object-based filtering schemes and conclude that object-based filtering is superior to grid-based filtering.

Macedonia *et al.* [10] describe a hexagonal grid-based approach to multicast and argue that hexagons are better than rectangles for use as grids. The authors also report that their NPSNET software is the first DIS simulation to use IP multicast for its communications.

Swaine and Stapf [11] discuss filtering at the source of data (called send-time filtering) and filtering in the network using multicast. They define a hierarchical filtering architecture that categorizes all the places in a distributed system where different types of filtering can occur. They go on to describe a filtering language made up of terms that are "and"ed together first then "or"ed, the opposite of Conjunctive Normal Form. Finally they suggest different multicast groups be used for different types of entities (especially aircraft).

Milgram [12] suggests PDUs be addressed to MC groups on the basis of entity type, PDU type, or geographic location.

Smith *et al.* [13] report on experimental results using ModSAF in which a geographical-grid multicast approach is used for Entity State PDUs (ESPDUs). Non-ESPDUs are sent on other MC groups. The authors then report on a new idea called *fidelity channels*, in which different multicast groups are used for Entity-State information of different resolutions. With this idea, an entity sends every *N*th ESPDU onto a low-resolution MC group, while all the rest are sent on a high-resolution MC group. This approach is used to ameliorate the wide-area viewer problem in which entities with large areas of interest could still be swamped with incoming data even when multicast is used. With fidelity channels, a wide-area viewer can subscribe to the low-resolution data and still have information spanning a large area, but not at the full update rate. To get the full update rate, a simulation must to subscribe to both fidelity channels, as data is not duplicated on either channel.

Calvin *et al.* [14], continuing the STOW work, report on progress implementing subscription agents. They go on to briefly mention two proposals for multicast group assignment: "First is the playbox grid scheme, which assigns a different multicast group to each grid square of a playbox. An alternate scheme using clustering based on multiple parameters such as fidelity requirements, classes of data, and geographic proximity could also be used." This statement represents the first mention of *data clustering*. Unfortunately, the authors do not give enough detail to enable the reader to understand what they mean by this term.

Pullen and White [15, 18, 21] describe a scheme called *dual-mode multicast*. In this scheme a geographic grid is used to define what multicast group to use on the LAN, while Exercise ID is used for multicast group selection on the WAN.

Aggarwal and Kelly [16] expand on the concept of fidelity channels, proposing a hierarchical fidelity structure allowing any number of levels of fidelity to be separated on different MC groups.

Though not really addressing multicast, Kerr and Dobosz [17] suggest filtering based on PDU type and suggest different UDP ports be used as the filtering mechanism.

The STOW Program continued to report on progress with filtering [19, 20, 22, 23] expanding on the concepts of geographic grid-based multicast and fidelity channels. They emphasize that a "physics-based" approach (in which sensor ranges are considered) was being used, rather than a "protocol-based" approach (in which only protocol information such as entity type is used). Again, clustering is briefly mentioned: "This approach forms clusters based on sets of parameters such as classes of data and geographic proximity of objects." The authors go on to say that clustering uses many fewer MC groups than geographic filtering, but that clustering is computationally too expensive. Unfortunately, again, the actual clustering algorithm is not described in enough detail for an independent evaluation. The authors also do a study on the effectiveness of grid-based filtering and show that it results in anywhere from a 50% to a 75% reduction in traffic for realistic (STOW-E) scenarios.

Powell *et al.* [24] write about the use of multicast in the JPSD-95 exercise. Simulations were arranged such that those with similar data requirements were grouped together on a single multicast group acting as a "virtual LAN." Gateways saw to it that only relevant data was transmitted between the "LANs" using a sophisticated interest declaration language based on predicates defined in Conjunctive Normal Form. The idea in this approach is *locality of reference*, the optimal grouping together of simulations based on the data they produce and their interests. Unfortunately, this scheme, though capable of being dynamically altered during runtime, was used only in a static fashion during the exercise.

These are all of the original multicast ideas in all of the DIS, CGF and I/ITSEC literature that I could find. In summary, the following ideas have been proposed as the basis for MC group assignment:

- PDU Type
- Exercise ID
- Radio Frequency
- Geographic grids of varying forms
- Entity Type
- Entity ID
- Fidelity Channel
- "Clustering" was mentioned though never fully explained
- Locality of reference

In addition, the concept of hierarchical filtering has been discussed and implemented in a number of systems. By far the most popular MC scheme is the first idea ever invented, the geographic grid scheme.

## 3.0 MULTICAST TAXONOMY

As can be seen, most of the schemes described above allocate multicast groups based on the value of a field inside the PDU. A more general categorization of the multicast allocation problem is required before we can understand alternatives to the schemes presented above. So when a simulation (or the RTI) is about to transmit a piece of data onto the network, it must decide what multicast group to put the data on. It has basically three pieces of information that it can use to make this decision:

1) The *source* of the data (i.e. the place the data is coming from: either site or host or process or unit or entity). This is called *source-based* multicast.
2) The *data* contents itself (i.e. what geographic location the data fits into, or what side the producing entity is on, or what markings it has, or what entity type it is, or what acceleration it has, etc.). Any multicast scheme where the multicast group is based on the data contents fits into this category. This is called *data-based* multicast.
3) The *destination* of the data. In *destination-based* multicast, the data is sent onto a multicast group solely based on those hosts that are to receive that information.

These are the only three pieces of information on which the decision may be based. Thus all multicast schemes must fit into one of these three categories or be a combination of these categories. I will now give brief examples of how each scheme might work in a flat topology. Note that the examples given below are just *some* of the ways multicast could be implemented for each of the categories described above. This discussion is not intended to be an exhaustive treatise on all the potential uses of multicast. Its purpose is to spark debate and research in the DIS and HLA communities.

**Source-Based Multicast.** In source-based multicast, every source of data is assigned its own multicast group, and every host then subscribes to only those sources it cares about. The problem then becomes: how do the destinations know about what information is being produced by each source? The answer lies in creating a special multicast group that everyone subscribes to that carries very low frequency information about all entities in the game (as in [9] or [13]). Since all hosts subscribe to this low-fidelity data channel, they know roughly where all the entities in the game are so they can then get more detail about the ones they really care about by subscribing to the multicast groups representing the sources they want to hear. One good thing about this scheme is that the number of multicast groups used scales linearly with the number of sources. In principle, the "source" can be anything, including a single entity. So in the limit of absurdity, every entity might have its own multicast channel on which to transmit (as in [5]). In this case, the number of multicast groups are high, but the number of unwanted packets arriving at destinations are very low, since each destination can address each entity exactly. A more reasonable approach would have one multicast group for each host that is transmitting. Then the number of multicast groups used would be smaller, but the risk of receiving unwanted data is greater (although not a lot greater as each host's entities tend to be geographically close to one another). Another advantage of this or any scheme with a low-fidelity channel is that it aids in solving the wide area viewer problem as discussed above. It also helps the "rapidly-steerable viewer" problem as there will always be (rough) data resident on each host for all the entities in the exercise.

The amount of bandwidth used by the low-fidelity channel need not be that great, as entities need only be updated on this channel when they move a significant distance. To give an example, if "a significant distance" is taken to be four kilometers, and entities move at an average rate of 20 km/hr, this means that every entity need be updated on the low fidelity channel every 12 minutes. For 100,000 entities in an exercise, this low-fidelity channel will produce roughly 140 PDUs/second globally, a significant amount, but not overwhelming. Of course, if the "source" is the host, then the number of global packets per second can be reduced significantly, as each host need only produce a single "Summary PDU" every once in a while describing where its entities are. Obviously there would be varying PDU production rates for the low-fidelity channel (more for fast-movers, less for infantry, etc.), but the calculation above shows that this scheme is at least plausible.

In source-based multicast, senders never change the multicast groups they send to, while receivers change subscriptions on an infrequent basis, as new sources come into range. Because of the existence of the low-fidelity channel, each simulation can take into account the time it takes to set up a MC connection in its decision about when to begin subscribing to newly relevant sources. Thus MC group set-up latency has little impact on the effectiveness of this scheme.

**Data-Based Multicast.** In data-based multicast, there are a number of different ways of mapping data to multicast groups, most of the more obvious ones having been discussed above. Almost all multicast proposals elaborated so far (including the ones being considered for the RTI) fit into this classification. The most obvious example is geographical filtering, where each piece of data is sent onto a multicast group based on the location of its originating entity, and each host subscribes to those multicast groups representing the geographical areas that the receiving host's entities are

interested in. There are obvious problems with geographic grids, the primary problem being that the required number of multicast groups scales as the area of the battlefield. If a grid of 3 km by 3 km is used (which is reasonable for ground forces), then for a 300 km x 300 km battlefield (which is relatively small), you would need 10,000 multicast groups. Of course, if the network hardware only supported 1,000 multicast groups, a 10 km x 10 km grid would be required, which would not be as optimal for data segmentation. Neither a large number of MC groups nor a large grid size is realistic in practice, so the geographic grid scheme will not, in general, work without some alterations.

To rescue geographic grids, you must throw away the fixed grid sizes and locations. Any number of schemes may be applied. As in [2], a topographically relevant partitioning of the battlespace might be used. Another alternative is a hierarchical grid scheme (such as quadtrees) in which more multicast groups are used where there are more entities and fewer groups are used in deserted parts of the battlefield. Another alternative would be to use an arbitrary mesh (like is used in large fluid-mechanics physics simulations). To make the arbitrary mesh scheme work, one computer somewhere on the network would need global information about all entities' positions. This computer, the "grid server" (or "subscription agent" in Calvin and Van Hook's terminology), would really only need low-fidelity information, so we again need a low-fidelity data channel. The grid server would continually analyze the battlefield with respect to all the entities' positions and calculate an optimal mesh of geographic regions that used a fixed number of multicast groups in a way that minimized the delivery of uninteresting data. The grid server would transmit the optimal grid to all of the computers on the network for their use. The grid server would then monitor the battle and judge when its continuously calculated optimal mesh becomes significantly different from the currently used mesh (i.e. some set threshold has been passed), then the grid server would send the new mesh out to all the simulation computers and the new mesh would then be used for transmission and subscription. Obviously some fancy protocols would be necessary to get the changeover to occur smoothly, but this problem is not insurmountable. The main problem in this approach is that the calculation of the optimal mesh is equivalent to the "set coverage" problem, a classic non-polynomial problem. Thus it cannot be solved at arbitrary scale in a computable amount of time. A heuristic optimization approach, such as simulated annealing, would be needed to get a good solution to this problem in computable time. The good news is that the mesh would change relatively infrequently (on the order of minutes or hours), thus significant computational resources could be devoted to solving the problem. Also, the previous mesh would probably be a very good first guess in calculating the new mesh.

The "Filter Space" concept included in the RTI Interface Specification [25] extends the geographical sectorization idea into more than simply two dimensions by potentially including all the other fields ("Attributes") in the PDU ("HLA Object") as independent axes on which to grid the data.[1] Obviously, the more axes you use, the worse this scheme scales with respect to the number of multicast groups required, if a fixed grid size is used. If the grid size is allowed to grow, of course, the filtering becomes less and less effective. So for example, just adding altitude to the two-dimensional example described above might required another factor of 10 in multicast groups: 100,000 multicast groups. Adding all the other fields in the Entity State PDU as separate dimensions gets you into the stratosphere rather quickly. Because of the practical limit on the number of multicast groups, extending the geographical sectorization scheme in this way is not viable.

Of course, the community has focused on geographic grids because geography has been the most promising method for eliminating unwanted data. The other discriminators mentioned in Section 2.0, such as PDU Type, Entity Type, etc., are all valid as well. In particular, assigning multicast groups on the basis of PDU type can be very effective since each PDU type is used in different ways by different simulations.

With data-based multicast, both senders and receivers of data need to change their multicast subscriptions, and unlike the source-based approach above, there is little forgiveness for networks that take time to add new members to MC groups. The reason for this is easily explained with a geographic grid example. As an entity moves from one grid cell to another, it *must* change its transmission from one MC group to another. Thus, for data-based MC to work, the time it takes to transmit to a new MC group must be less than the time between two PDUs issued from the same entity, a time that can be smaller than 0.1 seconds. On some network architectures this is no problem, and on others it is a problem. Subscription is much more forgiving, as simulations can anticipate their needs and take the join time into account just as in the source-based approach.

**Destination-Based Multicast.** Finally, we come to destination-based multicast. For this scheme to work, the simulation needs to calculate who is interested in any given PDU. Once a list of destinations is known, the data can be addressed to the multicast group that represents just those destinations. This scheme requires that the interest expressions of every simulation be transmitted to every other simulation during the exercise. These interest expressions change rather slowly, so they do not represent a large amount of information flowing between hosts compared to the simulation ground truth data. The interest expressions would be tagged with the originator of the interest, so when an outgoing PDU is compared to the list of currently active interest expressions, a list of valid destinations can be built. The number of multicast groups required in this scheme for a fully-connected mesh of simulation hosts is $2^N$-1, where $N$ represents the number of hosts in the exercise. Thus destination-based MC scales the worst of all the schemes discussed so far when it comes to the number of MC groups needed. Note, however, that this scheme, unlike the others, produces *perfectly segmented* data, in which no simulation

[1] Information other than explicit PDU contents can be used for filter space axes, but this does not change the essential nature of the filter space scheme.

would *ever* get *any* data it did not want. In this way the $2^N$ scaling represents the ideal case, to which other schemes are mere approximations. It is very important that everyone understand this point: destination-based filtering represents the best one can ever do at segmenting data, at the cost of some processing of remote interests and an enormous number of multicast groups (at least in a flat topology). Note that because interest expressions are resident on each host, hosts can identify PDUs no one is interested in and *not transmit them*, thus saving transmission costs and bandwidth (i.e. it facilitates source-filtering).

To save this scheme from failure without adding hierarchy (described below), it must be recognized that not all the $2^N$ multicast groups are going to be populated with data; only some pathways will actually be used. So to make this scheme more practical, each simulation would keep a fixed number of multicast groups defined for the most popular lists of destinations, and use a catch-all multicast group for any data that is not in this list. This method represents a *caching* technique, in which each of the most used multicast groups are cached and active, and when some become unused they can be dropped from the cache and replaced with other more active groups. This scheme makes sure that most of the data is segmented maximally, and that only data destined for the least-used combinations of destinations is not sent optimally. Any number of mechanisms exist for managing the cache, from a fixed number of MC groups per host to a global MC Cache Manager (again requiring global knowledge, but this time not of entities' positions, but of lists of sources and their most popular destinations).

Destination-based multicast is reasonably resilient to long join/leave times for MC groups. In the fully-connected case, of course, all MC groups are static. In the caching case, only when new groups are created (representing destination lists that are "swapped" into cache) will there be a potentially disruptive lag time. In these cases, the data that is affected can be sent on the catch-all group until the new groups are fully established. Thus, one can imagine a fairly benign effect of large join/leave times on destination-based MC, depending on the protocols used for managing the MC group cache.

Note also that destination-based schemes are inherently point-to-multipoint in nature and *not* multicast in nature, unlike the other schemes described above. In destination-based schemes the only things that matter for sending data are the source and the list of destinations, i.e. the point and the multipoint. This scheme could thus easily take advantage of the native ATM point-to-multipoint services with the concomitant increase in performance that native ATM has over standard IP multicast. To take advantage of this point-to-multipoint nature, this scheme must be implemented in an all ATM-to-the-desktop environment.

## 4.0 THE ROLE OF HIERARCHY

Each of the schemes above can also benefit from adding hierarchy to the data-distribution topology. Adding hierarchy means dividing up the computers used in an exercise into sets and adding a computer "middleman" or "gateway" to each set that: a) represents the outside world to the computers in its set, and b) represents the computers in its set to the outside world. Let's take an example of a simulation system that has 50 computers all representing active entities in an exercise. If we hooked all of these computers up together with no middlemen or gateways (a *flat* topology), each of these computers would in some fashion have to deal with information directly to and from all 49 other computers in the exercise. As we've seen above (especially in the source- and destination-based multicast schemes), dealing with large numbers of computers taxes the number of multicast groups greatly. Now imagine we divide the computers up into five sets of ten computers each, and add a gateway machine to each set. Now each computer in a set only has *ten* other computers it has to deal with, the nine other simulation computers in its set and the gateway computer. Note that from any simulation computer's perspective (call it Host A), the gateway is indistinguishable from another simulation computer. Host A sees entities generated by the gateway just as if it were a giant computer capable of simulating all the entities actually simulated by the other 40 simulation computers in the exercise. So as far as Host A is concerned, it is in an eleven-computer exercise. This can make each of the multicast methods described above easier to implement, because traffic can now be *segmented*, i.e. traffic generated in one set need never leave that set if no one outside of the set is interested in that data. It now behooves the exercise designer to segment the simulations in an optimal fashion so that simulations that care about each other's data are now in each other's set (as in [24]).

With hierarchy, each gateway receives an incoming packet from either external or internal sources and forwards it appropriately based on the multicast scheme used. It would be possible using hierarchy to have one multicast scheme used internal to a set, and a different one used external to the set (generalizing from [15]). In this case, the gateway knows about the various schemes and does the right thing depending on which way the data is heading. By intermixing schemes like this, one can deal with topologies that have a limited number of multicast groups either in a set or between sets. One can also deal with multiple network implementations such as ethernet LANs (the set), and an ATM WAN (the outside world). The gateway thus becomes not only a filter but a mechanism for "impedance matching" between two different network technologies.

But gateways can be useful even in a uniform network environment. With source-based multicast, it is possible that gateways could limit the number of multicast groups that are used globally (if the network supports the concept of locally-valid multicast groups). With data-based multicast, gateways are less obviously useful, but one could imagine that an optimally segmented network could eliminate a large amount of inter-gateway traffic and reduce WAN requirements under certain topological assumptions (such as when a WAN has much less bandwidth than the LANs). With destination-based multicast, hierarchy really pays off because now instead of the required number of multicast groups scaling like $2^N$-1, where $N$ is the total number of computers in the exercise, it scales like $P(2^M) + 2^P$, where

*M* is the number of computers in each set and *P* is the number of sets, a significant improvement. In our example above with the 50 computers, destination-based multicast requires $2^{50} \approx 1{,}000{,}000{,}000{,}000{,}000$ multicast groups in a flat topology and $5*2^{10} + 2^5 = 5{,}152$ in the five groups of ten topology. In a topology that has seven groups of seven computers, the required number of multicast groups (or point-to-multipoint connections) is $7*2^7 + 2^7 = 1{,}024$. It is obvious that hierarchy pays off big time in destination-based multicast.

Note that there are no limits to the number of hierarchical layers used in a simulation topology, as long as one is willing to pay the appropriate cost in latency. Each time a packet goes through a gateway, latency is added to the delivery of that packet. Keeping the latency down below some threshold related to the required fidelity of the exercise is very important. Yet this requirement does not exclude hierarchy from use, it simply requires it to be used appropriately and in moderation. Note that a strict hierarchy is not necessary. Some simulation hosts might communicate to their own gateway while others communicate directly to the set of other simulation gateways. Some sets may be further subdivided into subsets, each with its own gateway, and some not divided, etc.

## 5.0 MILITARY UNIT-BASED MULTICAST

Of the multicast schemes addressed above, the one deserving a more detailed explanation is the source-based approach, as this approach has the possibility of achieving a high degree of segmentation without using a large number of multicast groups. The important point is that we are in the business of *military combat simulation*, and we can exploit the natural arrangement of military forces to help segment simulation traffic. This idea represents an extension of the idea by Calvin *et al.* [19] to segment traffic based on information from the virtual world, as opposed to basing it on protocol-oriented information. At first the discussion will be restricted to Entity State PDUs. Other PDU types will be discussed later.

Of all the potential "sources" that could be used for determining multicast group selection, the one that has gotten very little attention but could yield great benefits is the *military unit* of the transmitting entity. This approach, a type of source-based multicast, is called *Unit-Based Multicast* (UBM). The basic idea is that each simulation transmits each entity's state on a multicast group based on the unit that the entity belongs to. The echelon on which this segmentation occurs need not be fixed across the exercise: it could be companies for some forces, battalions for others, brigades, etc. In general, the decision about what echelon is the right one to use for a given simulation depends on the exercise requirements and composition, the types of simulations used, and the number of entities each simulation can host on any given computer. UBM, like all source-based MC schemes, requires a global low-fidelity MC group on which each simulation would transmit *unit-level* summary information. That is, with UBM the low-fidelity MC group is used for (modified) Aggregate State PDUs representing the units that are the basis for the multicast segmentation. The Aggregate State PDU would contain information on the unit's geographic area of operations so that potential subscribers could judge whether the unit is relevant to them or not. It would also contain the MC address where the more detailed information could be found. The number of PDUs/second on the low-fidelity MC group would be small as this aggregate-level information needs to be updated very infrequently, only when significant changes occur to the forces represented in the aggregates. Every simulation would listen to the low-fidelity MC group and decide which units are of interest, and then subscribe to the appropriate units' MC groups.

There are a number of advantages to this approach. First, we exploit the fact that entities on the same computer are generally in the same unit (at some echelon). This means that, in general, hosts need to transmit on a limited number of MC groups. The best solution would be to pick the MC segmentation echelon such that any given simulation host transmits on a very few or even a single MC group. The echelon chosen for segmentation need not be confined to a single simulation running on a single computer. For instance, if a battalion's entities are spread over two computers, a single MC group could be used for that battalion as long as only one of the computers controls and transmits the battalion-level information on the low-fidelity MC group. The number of MC groups needed for an exercise then scales with the number of computers used in the exercise, a very benign scaling. Second, the MC groups used for transmission are constant during an exercise. Third, units generally are only in contact with a small number of other units, thus limiting the number of MC groups a simulation needs to subscribe to. Also, the units in contact change very infrequently, thus mitigating any join/leave latency inherent in the underlying MC mechanism. Fourth, entities in the same military unit are generally in the same geographic region, so this scheme gives a good geographic segmentation without explicitly using geography (with all its attendant baggage) as a segmentation mechanism. Fifth, because aggregate information is transmitted over the low-fidelity channel, all simulations have significant knowledge of all military units in an exercise. This benefits simulations or plan-view displays that really only need aggregate-level information most of the time, and only need entity-level information once in a while. Basically, you get the advantages of using broadcast, without the large volume of network traffic. Sixth, this scheme represents a mechanism for seamlessly adding aggregate simulations to an exercise. When no other simulations are interested in entity-level detail, aggregate simulations can operate in an aggregate mode. When some simulation does require more detail, the aggregate simulation can deaggregate or transfer control of its forces to an entity-level simulation.

The UBM idea also has a number of problems that require more discussion. First, how does the scheme deal with a unit's munition entities that come into existence, fly for a while, and then land a long way away from the originating unit? The scheme quickly falls apart if munition entities transmit on the same MC group as their shooters. The essential idea behind UBM segmentation is for a unit's aggregate information on the low-fidelity channel to indicate a geographically

small area of operations to maximize segmentation. If the unit's area of operations needs to be expanded to include the effects of any possible munition entity created and launched, the area would be so large that the MC segmentation would be useless. To solve this problem, more MC groups are needed for munition entities. Each simulation must transmit additional aggregate information on the low-fidelity channel. As well as each unit's area of operations, the simulation must transmit the area of operations of the munition entities potentially launched by those units as well.

An example will clarify this approach. Suppose a given computer simulates two battalions: an MLRS Battalion and a Mechanized Infantry Battalion. This simulation (Sim A) would transmit its information onto four separate MC groups. MC Group 0 would be the low-fidelity channel. On it, the simulation would occasionally transmit Aggregate State PDUs containing the area of operations for each battalion as well as the address of the MC group where the detailed information about these units is to be found. The Aggregate State PDU for the MLRS Battalion would contain additional information about the potential area of operations for the MLRS rocket entities, as well as the address of the multicast group where this battalion's MLRS rockets' ESPDUs would be found. On MC Group 1, the simulation would transmit all ESPDU traffic for the Mechanized Infantry Battalion's entities. On MC Group 2 the simulation would transmit all ESPDU traffic for the MLRS Battalion's entities. On MC Group 3, the simulation would transmit all ESPDU traffic for MLRS rocket entities. Thus a subscribing simulation (Sim B) would always subscribe to Group 0 and get the aggregate view. If any of Sim B's entities were close enough to either the Mech Battalion or the MLRS Battalion to be able to directly sense their entities, Sim B would subscribe to MC Groups 1 or 2 respectively. If Sim B's entities were within range of the MLRS rockets (and included some means of detecting or targeting the rockets as entities), Sim B would subscribe to MC Group 3. Of course, if Sim B's entities were in range of the MLRS rockets but did not have any means of sensing or detecting the rockets, it would *not* subscribe to MC Group 3. (Remember, only ESPDUs are being considered here, Detonation PDUs are covered next.)

A similar problem occurs with some interactions. Fire and Collision PDUs are coincident with the issuing entities' positions, so these PDUs could be sent on the same MC groups as the ESPDUs for a given unit. (On the other hand it might be useful to segment these PDUs by PDU type as well. A more detailed analysis is needed to see which scheme would be best.) Detonation PDUs, on the other hand, fall into the same category as munition entity ESPDUs described above, as they can occur a long way away from a given unit's position. Thus the solution for Detonation PDUs is identical to the solution for munition ESPDUs: a summary of a unit's potential detonation area is sent on the low-fidelity channel (as part of the unit's Aggregate State PDU), and a separate MC group is created for a given unit's Detonation PDUs.

Dynamic environment objects (such as smoke clouds) are treated in the same way: aggregate summaries are sent on the low-fidelity channel, and another MC group is reserved for high fidelity environment information. The principle for almost every other type of information is the same, aggregate information on the low-fidelity channel and details on a separate MC group. The only exceptions to this methodology would be simulation management information (sent on a single global MC group) and radio signal information (segmented onto MC groups on the basis of frequency). Using the UBM scheme, any simulation would transmit on only a few MC groups depending on what type of information it produced. Simple simulations that just simulated ground units might only transmit on a few groups: the low-fidelity group, the simulation's unit group, and the detonation group. More sophisticated simulations that use radios, munition entities, and dynamic environment entities would need to transmit on more groups, but the total would be limited.

Another potential problem is that the UBM scheme does require more effort on the part of simulations: they now need to transmit, understand, and track unit-level information as well as entity-level information. More analysis is needed to determine how much additional processing is required. It will also be more difficult to retrofit legacy simulations into this scheme since most do not keep track of both aggregate-level and entity-level views of the battlefield. New simulations, however, such as WARSIM 2000, NASM, JSIMS, etc. should have no problem adopting a scheme like UBM.

This method has clear advantages over the popular geographic grid scheme. Primarily, it requires only a handful of MC groups for each computer in an exercise. For an exercise of 20,000 entities that uses 100 computers (200 entities/computer), a reasonable estimate is seven MC groups per computer (three different MC groups for three units' entity state, collision, emission, and fire PDUs, one MC group for all three units' detonations, one for all three units' munition entities, and one for dynamic environment entities) plus another hundred or so global MC groups (for weather data, signal traffic, simulation management traffic, and the low-fidelity channel) leading to a total of 700 MC groups for a very large exercise. This number of MC groups is reasonably consistent with the hardware limitations of both ethernet and ATM, and so should lead to excellent segmentation.

The UBM scheme was proposed for use in the STOW program, but was rejected in favor of the Filter Space scheme [25] and its (apparently fixed) grid-based approach. Nevertheless, I believe the UBM method shows real promise because it bases its multicast segmentation on *militarily-relevant information* (unit organizations), information that ought not to be ignored if a large-scale *military* simulation is the goal.

## 6.0 SUGGESTIONS FOR FUTURE MULTICAST EFFORTS FOR DIS AND HLA

This paper has presented a few different multicast schemes that would help minimize the number of undesired PDUs showing up on simulation hosts. Of the three most promising techniques described above (unit-based, optimal-mesh data-based, and hierarchical cached destination-based), I have *no opinion* about which one

would be the best scheme. No data exists comparing these methods to each other. Also, each scheme may work better or worse depending on which simulation, computer hardware, underlying network technology, or combination of technologies are used. When all the variables that need to be analyzed are added up, it is obvious that deciding which is the best scheme would require significant research. It may be that one scheme is best under some conditions, but not as good under other conditions.

It would be a mistake for the simulation community to standardize on *any* multicast scheme in the absence of any evidence of how the scheme performs. What is required is a strong research effort on all of the different ways to use multicast comparing and contrasting them under all sorts of realistic operational conditions. Only then will enough data be gathered so that future exercises and programs will be able to make informed decisions about how to use multicast, filtering, and interest management to meet their needs. Even then, I predict that no one scheme will be a clear overall winner under all circumstances. Combat simulation is too wide a field and has too many variables to imagine that there exists a "one size fits all" solution.

Thus, the only thing the DIS and HLA community can do is be prepared to use a number of different interest management and MC schemes under different conditions to meet different requirements. The question then becomes one of how to accomplish this goal. One must look at all of the most promising multicast schemes mentioned above to understand the only possible solution. In all three schemes, the multicast segmentation is accomplished by having a deep understanding of what is happening operationally in the exercise. In the unit-based multicast case, the simulations need to understand the concept of units as aggregates of entities and be able to relate what the units are doing to their own entities' data requirements. In the optimal-mesh data-based scheme, the mesh server must understand fully the data requirements of all the exercise's entities, so it can build an optimal mesh. This means it must understand sensor ranges, entity move rates, everything. In the hierarchical cached destination-based scheme, each simulation must know about the interests of all the other simulations, and be able to evaluate these interests against outgoing data. So for all of the most promising multicast techniques, the decision about what multicast group a particular PDU ("HLA Object") gets transmitted on requires intimate knowledge of the simulation, the exercise, and the contents of the PDU. What must be done now becomes clear. The RTI, by explicit HLA rules, is required *not* to know any of this information, including how to interpret any given attribute (field) in any given object (PDU). So either this rule must be changed and the RTI be allowed to understand the contents and meaning of the objects it ships around, or multicast group selection must be made *outside* the RTI. The first option being impractical, we are left with only one choice: *filtering decisions and multicast group selection must be made in the federate.* Only the simulation has enough information to make the sophisticated decisions necessary to make the most promising multicast schemes work. The RTI interface would then be very simple. Each time a set of attributes are updated or an interaction is sent, an *address* parameter would be supplied as part of the appropriate function telling the RTI which multicast group to use for an object. Thus,

```
oneway void update_attribute_values(
  in ObjectId objectID,
  in AttributeNameValuePairSet attributeList,
  in FederationTime theTime);
```

becomes:

```
oneway void update_attribute_values(
  in ObjectId objectID,
  in AttributeNameValuePairSet attributeList,
  in FederationTime theTime,
  in Address theAddress);
```

with a similar change for `send_interaction()`. An additional function such as:

```
oneway void subscribe_address(
  in Address theAddress);
```

would be needed to tell the RTI which MC groups to join for subscription purposes. These changes (along with defining exactly what an `Address` is) would be the only changes needed in the RTI API to ensure that the RTI implements a general approach to filtering and multicast. I call this the *Address Filtering API* (AFA).

The current Filter Space approach is based on attempting to give the RTI hints about the meaning of the objects it transmits without breaking the rules. Unfortunately, its convoluted nature will necessarily lead to interoperability problems as federates misuse the API. The API and its single implementation also backs the community into a single mindset about filtering that hampers development of alternative interest management or multicast techniques. We should remember, however, that any multicast technique can be built underneath any reasonably expressive API, including Filter Space. One could, for instance, very easily use the C library function `printf()` as your multicast API. But there would be no benefit in such an approach. Because the semantics of using the API necessarily change depending on the filtering technique used, no two simulations could interoperate unless they were using the same semantics and thus the same filtering scheme. There is no reason therefore to define such a high-level API as Filter Space, without first defining a lower-level addressing API like the AFA. The AFA would be much simpler to implement, much more effective at promoting interoperability, and much more compatible with any and all interest management and multicast schemes that are invented in the future. If desired, a Filter Space implementation could even be built on top of the AFA. Since the AFA is completely general, any number of increasingly sophisticated filtering packages (each with their own tailored API) could be built on top of it. These filtering packages would use the AFA, but give the federates significant added interest management capability. Because these filtering packages would not be part of the core RTI, the requirement for not understanding the data passed by the RTI could be relaxed, so these filtering packages could implement some of the sophisticated schemes described

in Sections 3.0, 4.0, and 5.0. The basic idea behind this approach is *modular* software architecture, design, and engineering, in which a layered architecture is used to separate out the low-level functionality (such as addressing) into one layer and the higher-level functionality (such as interest management) into a higher layer, with each layer being isolated from the next by a well-defined interface appropriate to the layer's purpose. Adding the AFA will increase the RTI's flexibility, extensibility, scalability, and most importantly, *interoperability*, all qualities crucial for the success of the HLA.

In conclusion, the work the DIS community has done so far in inventing new ways of using multicast has been good, but much too focused on the geographic grid scheme and its derivatives. A number of new ideas for using multicast have been presented in this paper in an attempt to spark debate and research in the DIS and HLA communities. The role of interest management and multicast group selection has been proven to be best done outside the RTI. And finally, small additions to the RTI API have been suggested that give the RTI a general addressing mechanism on which any number of different interest management and multicast schemes could be built, not just the one Filter Space scheme. Given that scalability is one of the most important issues facing the distributed simulation community, and that interest management and multicast represent the best hope for achieving a scalable simulation architecture, my basic conclusion is that we in the DIS community must solve these problems the right way, based on research, experiments, and data. And until the issue is decided of which are the best, most scalable multicast schemes, we should not lock ourselves into a single solution that impedes the insertion of new multicast algorithms when they are invented. Only by using proven modular software architecture techniques along with our best ideas and inventions can we achieve the goal of building a truly scalable distributed simulation system.

## 7.0 ACKNOWLEDGMENTS

The author would like to thank Larry Mellon, Darrin West, Jesse Aronson, Ben Wise, Glenn Tarbox, Jim Watson, Josh Nan, Rich Briggs, Jack Harrington, and Tim Eller for many profitable discussions about the role of multicast and interest management in distributed simulation. Many of the ideas discussed in this paper were the result of collaboration between the author and these individuals and their contributions are gratefully acknowledged.

## 8.0 REFERENCES

[1] M. Johnson and S. Myers, "Allocation of Multicast Message Addresses for Distributed Interactive Simulation," **Proceedings of the 6th DIS Workshop**, IST-CR-92-2, 3/92, p. 109.

[2] R. Sherman and B. Butler, "Segmenting the Battlefield," **Proceedings of the 7th DIS Workshop**, IST-CR-92-17.1, 9/92, p. A-27.

[3] B. Clay, "Multicast or port usage to provide hierarchical control," **Proceedings of the 8th DIS Workshop**, IST-CR-93-10.1, 3/93, p. A-3.

[4] K. Doris, "Issues Related to Multicast Groups," **Proceedings of the 8th DIS Workshop**, IST-CR-93-10.2, 3/93, p. 279.

[5] Minutes of the Communications Architecture and Security Subgroup, **Proceedings of the 9th DIS Workshop**, 9/93, pp. 298-300, 359-366

[6] D. Van Hook *et al.*, "Scaleability Tools, Techniques, and the DIS Architecture," **Proceedings of the 15th I/ITSEC Conference**, 11/93, p. 837.

[7] D. Van Hook *et al.*, "An Approach to DIS Scalability," **Proceedings of the 11th DIS Workshop**, IST-CR-94-02, 9/94, p. 347.

[8] J. Calvin and D. Van Hook, "AGENTS: An Architectural Construct to Support Distributed Simulation," **Proceedings of the 11th DIS Workshop**, IST-CR-94-02, 9/94, p. 357.

[9] D. Van Hook *et al.*, "Approaches to Relevance Filtering," **Proceedings of the 11th DIS Workshop**, IST-CR-94-02, 9/94, p. 367.

[10] M. Macedonia *et al.*, "Exploiting Reality with Multicast Groups: A Network Architecture for Large Scale Virtual Environments," **Proceedings of the 11th DIS Workshop**, IST-CR-94-02, 9/94, p. 503.

[11] S. Swaine and M. Stapf, "Large DIS Exercises - 100 Entities Out Of 100,000," **Proceedings of the 16th I/ITSEC Conference**, 11/94, p. 4-13.

[12] D. Milgram, "Strategies for Scaling DIS Exercises Using ATM Networks," **Proceedings of the 12th DIS Workshop**, IST-CR-95-01.1, 3/95, p. 31.

[13] J. Smith *et al.*, "Prototype Multicast IP Implementation in ModSAF," **Proceedings of the 12th DIS Workshop**, IST-CR-95-01.1, 3/95.

[14] J. Calvin *et al.*, "STOW Real-Time Information Transfer and Networking System Architecture," **Proceedings of the 12th DIS Workshop**, IST-CR-95-01.1, 3/95, p. 343.

[15] J. Pullen and E. White, "Dual-Mode Multicast for DIS," **Proceedings of the 12th DIS Workshop**, IST-CR-95-01.1, 3/95, p. 505.

[16] S. Aggarwal and B. Kelly, "Hierarchical Structuring for Distributed Interactive Simulation," **Proceedings of the 13th DIS Workshop**, IST-CR-95-02, 9/95, p. 125.

[17] R. Kerr and C. Dobosz, "Reduction of PDU Filtering Time Via Multiple UDP Ports," **Proceedings of the 13th DIS Workshop**, IST-CR-95-02, 9/95, p. 343.

[18] J. Pullen and E. White, "Analysis of Dual-Mode Multicast for Large-Scale DIS Exercises," **Proceedings of the 13th DIS Workshop**, IST-CR-95-02, 9/95, p. 613.

[19] J. Calvin *et al.*, "Data Subscription,"

**Proceedings of the 13th DIS Workshop**, IST-CR-95-02, 9/95, p. 807.

[20] J. Calvin *et al.*, "Data Subscription in Support of Multicast Group Allocation," **Proceedings of the 13th DIS Workshop**, IST-CR-95-02, 9/95, p. 593.

[21] J. Pullen and E. White, "Simulation of Dual-Mode Multicast Using Real-World Data," **Proceedings of the 14th DIS Workshop**, IST-CR-96-02, 3/96, p. 507.

[22] S. Rak and D. Van Hook, "Evaluation of Grid-Based Relevance Filtering for Multicast Group Assignment," **Proceedings of the 14th DIS Workshop**, IST-CR-96-02, 3/96, p. 739.

[23] D. Van Hook *et al.*, "Performance of STOW RITN Application Control Techniques," **Proceedings of the 14th DIS Workshop**, IST-CR-96-02, 3/96, p. 1025.

[24] E. Powell *et al.*, "Joint Precision Strike Demonstration (JPSD) Simulation Architecture," **Proceedings of the 14th DIS Workshop**, IST-CR-96-02, 3/96.

[25] Defense Modeling and Simulation Office, **The RTI Interface Specification 0.5,** March 1995. Available from http://www.dmso.mil/projects/hla.

## 9.0 ABOUT THE AUTHOR

**DR. EDWARD T. POWELL** is a senior scientist with Science Applications International Corporation. He received his Ph.D. in Astrophysics from Princeton University and was the chief simulation infrastructure architect for the largest operational warfighter DIS exercise ever created, the 8,300-entity live/virtual/constructive JPSD-95 exercise. He is now a lead architect on the ARPA-funded Synthetic Theater of War Program.